ARMY RESEARCH LABORATORY

Parallel Performance of Linear Solvers and Preconditioners

by Joshua C. Crone and Lynn B. Munday

ARL-TR-6778 January 2014

Approved for public release; distribution is unlimited.

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory
Aberdeen Proving Ground, MD 21005-5066

ARL-TR-6778 **January 2014**

Parallel Performance of Linear Solvers and Preconditioners

Joshua C. Crone and Lynn B. Munday
Computational and Information Sciences Directorate, ARL

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)* January 2014	**2. REPORT TYPE** Final	**3. DATES COVERED (From - To)** February 2013–August 2013

4. TITLE AND SUBTITLE Parallel Performance of Linear Solvers and Preconditioners	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Joshua C. Crone and Lynn B. Munday	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army Research Laboratory ATTN: RDRL-CIH-C Aberdeen Proving Ground, MD 21005-5066	**8. PERFORMING ORGANIZATION REPORT NUMBER** ARL-TR-6778
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

In this report we examine the performance of parallel linear solvers and preconditioners available in the Hypre, PETSc, and MUMPS libraries to identify the combination with the shortest wall clock time for large-scale linear systems. The linear system of equations in this work is produced by a finite element code solving a linear elastic boundary value problem (BVP). The boundary conditions for the linear elastic BVP are produced by a discrete dislocation dynamics (DDD) simulation and change with each timestep of the DDD simulation as the dislocation structure evolves. However, the coefficient—or stiffness matrix—remains constant during the DDD simulation and some expensive matrix factorizations only occur once during initialization. Our results show that for system sizes of less than three million degrees of freedom (DOF), the MUMPS direct solver is 20× faster than the best iterative solvers per timestep, but has a large upfront cost for the LU decomposition. Systems larger than three million DOFs require iterative solvers. The Hypre algebraic multigrid (AMG) preconditioner packaged was the best performing iterative solver, but was found to be sensitive to the AMG parameters. The PETSc Block Jacobi preconditioner showed good performance with the default preconditioning setting.

15. SUBJECT TERMS

parallel, linear solvers

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Joshua C. Crone
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	UU	24	**19b. TELEPHONE NUMBER** (*Include area code*) 410-306-2156

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures **iv**

List of Tables **v**

1. Introduction **1**

2. Linear Elasticity Problem Statement **2**

3. Description of Linear Solvers and Preconditioners **2**

4. Results and Discussion **3**

4.1 Iterative Solvers..4

4.2 Direct Solver..10

5. Conclusions **12**

6. References **13**

List of Symbols, Abbreviations, and Acronyms **14**

Distribution List **15**

List of Figures

Figure 1. Relative serial performance of various solver – preconditioner combinations for mesh no.1. Values are compared to the PETSc solver without preconditioning......................5

Figure 2. Relative serial performance of various solver – preconditioner combinations for mesh no. 2. Values are compared to the PETSc solver without preconditioning.....................6

Figure 3. Parallel performance of Hypre – Jacobi and Hypre – AMG for two sets of AMG parameters. Ideal refers to the solve time of Hypre – Jacobi assuming 100% parallel efficiency..7

Figure 4. Parallel performance of various solver – preconditioner combinations with mesh no. 2. Ideal refers to the solve time of Hypre – AMG assuming 100% parallel efficiency.......9

Figure 5. Parallel performance of various solver – preconditioner combinations with mesh no. 3. Ideal refers to the solve time of Hypre – Jacobi assuming 100% parallel efficiency....10

Figure 6. Comparison of direct solver with various iterative solver – preconditioner combinations. Both solve time and setup time for MUMPS are included. Ideal refers to the solve time of Hypre – AMG assuming 100% parallel efficiency....................................11

List of Tables

Table 1. List of solver packages and corresponding preconditoners used in this work..................3

Table 2. Number of elements and total degrees of freedom (DOF) for the three meshes used in this work. 4

Table 3. Parametric study of the Hypre – AMG parameters to optimize serial performance. The bottom row corresponds to the final set of parameters referred to as “serial optimized params” throughout this work...6

Table 4. Parametric study of the Hypre – AMG parameters to optimize parallel performance. The bottom row corresponds to the final set of parameters referred to as “parallel optimized params” throughout this work...8

Table 5. Comparison of Hypre – AMG and MUMPS. 12

INTENTIONALLY LEFT BLANK.

1. Introduction

The need to effectively and efficiently solve linear systems of equations is an important computational challenge affecting a wide range of applications in scientific computing from solid mechanics and quantum mechanics to climate modeling and computational geometry. A linear system of equations can be represented as

$$A_{ij}x_j = b_i \tag{1}$$

where A_{ij} is the coefficient matrix, b_i is the right-hand side vector, and x_i is the vector of unknowns to be solved for. In this report, we solve the linear system of equations related to the three-dimensional (3-D) linear elastostatic boundary value problem (BVP). The focus on 3-D linear elasticity is motivated by our effort to accurately capture free-surface effects in discrete dislocation dynamics (DDD) simulations. This can be achieved by coupling a DDD simulator for bulk material (Arsenlis et al., 2007) to a finite element method (FEM) code that computes the image stress field resulting from the presence of free surfaces at each timestep. We have developed a massively parallel FE code, FED3, to perform the free-surface computations (Crone et al., 2013). A typical DDD simulation in bulk will require many timesteps (on the order of 1e5 to 1e6) to reach the desired loading conditions. To achieve similar simulation time scales with free surfaces and highly refined FEM meshes requires extremely efficient and scalable linear solvers to compute image stresses with a wall-clock time of a few seconds (s) or less. The keys to reducing solve time is to minimize parallel communication, computational expense per iteration, and number of iterations.

In this discrete dislocation dynamics-finite element method (DDD-FEM) coupling application, the coefficient matrix (stiffness matrix) remains constant from one timestep to the next and only the right-hand side vector (due to changing boundary conditions) needs to be updated. Therefore, the setup time for the linear solver and preconditioner only occur on the first solve. For this reason, we restrict our solver performance study to the time spent solving the system of equation and ignore the setup costs—except in the case of direct solvers where the setup cost can become prohibitively expensive for large systems.

In this report, we present a brief description of the problem we are solving as well as the linear solvers and preconditioners employed for this study. We go on to present the performance results for each of the solver – preconditioner combinations for various system sizes in terms of both serial performance and parallel scalability. We conclude by comparing the relative advantages and drawbacks of the solver – preconditioner options. We emphasize that while we restrict our evaluation to solving 3-D linear elasticity in this report, the results from this study can be applied to a wide range of applications where the solution of a linear system of equations is required.

2. Linear Elasticity Problem Statement

The formal statement of the 3-D linear elastostatic BVP is as follows (using index notation with Einstein summation convention):

$$\sigma_{ij,j} = 0 \; in \; \Omega \tag{2}$$

$$u_i = u_o \; on \; \Gamma_u \tag{3}$$

$$\sigma_{ij} n_j = T_o \; on \; \Gamma_T \tag{4}$$

where σ_{ij} is the stress tensor, $\sigma_{ij,j}$ is the divergence of the stress tensor, u_i is the displacement field, u_o is the prescribed displacement field on the surface Γ_u with Dirichlet boundary conditions, n_j and T_o are the surface normal and prescribed tractions, respectively, on surface Γ_T with Neumann boundary conditions, and Ω is the volume of the linear elastic body. The stress tensor is defined in terms of the infinitesimal strain tensor (ε_{kl}) by the generalized Hookes law:

$$\sigma_{ij} = C_{ijkl} \epsilon_{kl} \tag{5}$$

where C_{ijkl} are the elastic coefficients and ε_{kl} can be further defined in terms of the displacement field:

$$\varepsilon_{kl} = \frac{u_{k,l} + u_{l,k}}{2} \tag{6}$$

Substituting equation 5 and 6 into equation 2 and converting to the weak form (see Hughes, 2000 for detailed derivation of the weak form and implementation into the FEM framework), we recover equation 1 where A_{ij} is the stiffness matrix, x_j is replaced by the unknown displacement field values (u_j), and b_i is the force vector containing the contributions of the Dirichlet and Neumann boundary conditions. The resulting stiffness matrix is symmetric positive definite (SPD); therefore, we focus our study on conjugate gradient (CG) solvers, which are best suited for SPD matrices (Shewchuk, 1994).

3. Description of Linear Solvers and Preconditioners

For this work, we examine the performance of three open-source, parallel linear solver libraries and a fourth linear CG solver developed at the U.S. Army Research Laboratory (ARL). The first open-source library is Hypre, a suite of parallel iterative solvers and preconditioners developed by Lawrence Livermore National Laboratory (LLNL) (Falgout and Yang, 2002). The second open-source library is the Portable, Extensible Toolkit for Scientific Computation (PETSc), which is a combination of data structures, linear solvers, and preconditions developed by Argonne National Laboratory (ANL) (Balay, 2013). PETSc has a built-in interface for many external solver packages, including the third solver library—the Multifrontal Massively Parallel Sparse Direct Solver (MUMPS)—which is a parallel direct solver (Amestoy et al., 1998). The fourth linear solver is a matrix-free iterative CG solver developed at ARL during testing. The matrix-free form reduces the parallel communication and therefore displays excellent parallel scalability. However, it is much less efficient than the other libraries and only has a simple Jacobi preconditioner implemented. It is included as a benchmark when examining parallel scalability.

A major benefit of PETSc and Hypre is the suite of available preconditioners that can be used to reduce the number of solver iterations. Table 1 contains a list of all preconditioners used for each solver. Additional information about each preconditioner can be found in the reference manuals packaged with each library.

Table 1. List of solver packages and corresponding preconditoners used in this work.

Library/Solver	Preconditioner
PETSc	None
	Jacobi
	Block Jacobi
	Additive Schwarz (ASM)
	Geometric Algebraic Multigrid (GAMG)
	Incomplete LU Factorization[a] (ILU)
	Incomplete Cholesky Factorization[a] (ICC)
	Successive Over Relaxation[a] (SOR)
Hypre	Jacobi
	Algebraic Multigrid (MLI)
	Algebraic Multigrid (AMG)
	Parasails
MUMPS	None (Direct Solve)
Matrix-Free CG	Jacobi

[a]Indicates serial preconditioners.

4. Results and Discussion

To evaluate the performance of the various linear solvers and preconditioners, we model uniaxial tension of an elastic brick. The X, Y, Z dimensions of the brick are 21.8 × 8.175 × 8.175 µm, with a distributed force in the X direction of 100 MPa/m^2. Three quadratic tetrahedral meshes are created to study the effect of system size on solver performance. Table 2 contains the respective sizes of each mesh.

Table 2. Number of elements and total degrees of freedom (DOF) for the three meshes used in this work.

Mesh	Elements	DOFs
1	9279	42,057
2	72,120	308,661
3	5,284,709	22,414,125

All simulations have been carried out on the Pershing Supercomputer at the ARL Department of Defense (DOD) Supercomputing Resource Center (ARL-DSRC). Pershing is an IBM* iDataPlex containing two Intel† Sandy Bridge 8-core processors and 32 GB of memory per node. The compute nodes are interconnected by FDR-10 InfiniBand.‡

4.1 Iterative Solvers

We begin by comparing the iterative solver times for meshes no. 1 and no. 2 in serial to establish the optimal solvers when parallel communication is not involved. The solve time and number of iterations for each solver is measured relatively to the PETSc solver with no preconditioner. Without preconditioning, mesh no. 1 solves in 7.5 s and 1152 iterations, while mesh no. 2 solves in 145 s and 2341 iterations. The relative improvements of the various solver – preconditioner combinations are included in figures 1 and 2.

There are two timings for the Hypre – AMG to demonstrate the sensitivity of the preconditioning parameters parameters. Other works have shown AMG to be an efficient method for solving linear elasticity (Baker et al., 2009), but the performance is closely coupled to the selection of proper parameters. The Hypre – AMG preconditioner (BoomerAMG) provides a convenient interface for selecting the appropriate parameters. As shown in figure 1, by using the default parameters, BoomerAMG takes approximately 30% longer than PETSc – None to solve. However, after performing a parametric study on the BoomerAMG parameters, shown in table 2, we are able to reduce the solver time by a factor of three. The set of parameters determined from the study in table 2 are labeled as "serial optimized params" in figures 1 and 2. The label "parallel optimized params" in figure 2 is determined through a second parametric study, which optimized the BoomerAMG parameters for parallel performance. Further details of this study are included later in this section. For a description of the complete set of BoomerAMG parameters, we refer the reader to the Hypre manuals included with the library.

The iterative solver – preconditioner combinations with the shortest solver time for both mesh no. 1 and no. 2 are the PETSc – Block Jacobi, PETSc – ILU, and Hypre – AMG. While the relative reduction in solver time is constant between meshes no. 1 and no. 2 for PETSc – Block Jacobi and PETSc – ILU, the reduction in solver time for Hypre – AMG improves as we increase

* IBM is a registered trademark of the International Business Machines (IBM) Corporation, United States.

† Intel is a registered trademark of the Intel Corporation.

‡ InfiniBand is a registered trademark of the InifiniBand Trade Association (IBTA).

the system size. These results suggest that Hypre – AMG has the best performance for serially solving a system of equations when the AMG parameters have been carefully selected.

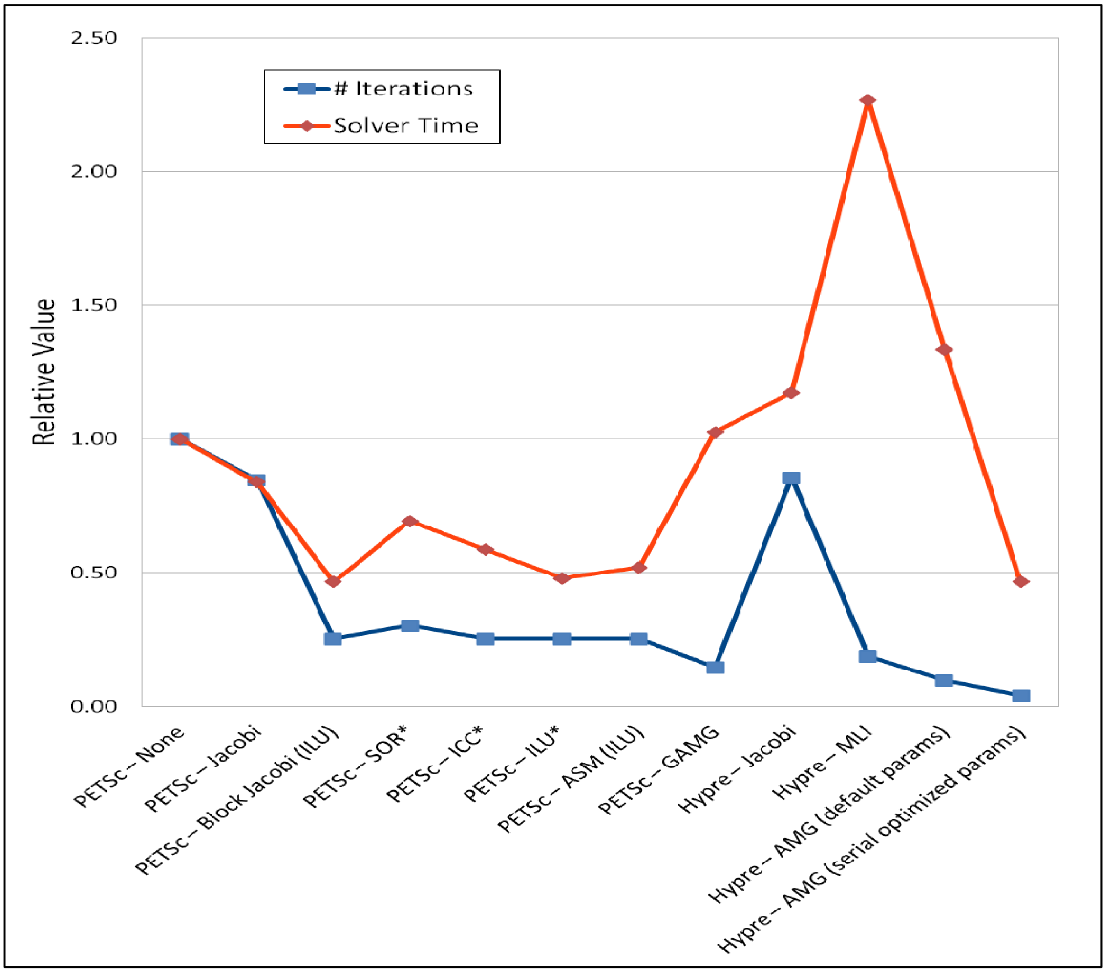

Figure 1. Relative serial performance of various solver – preconditioner combinations for mesh no.1. Values are compared to the PETSc solver without preconditioning.

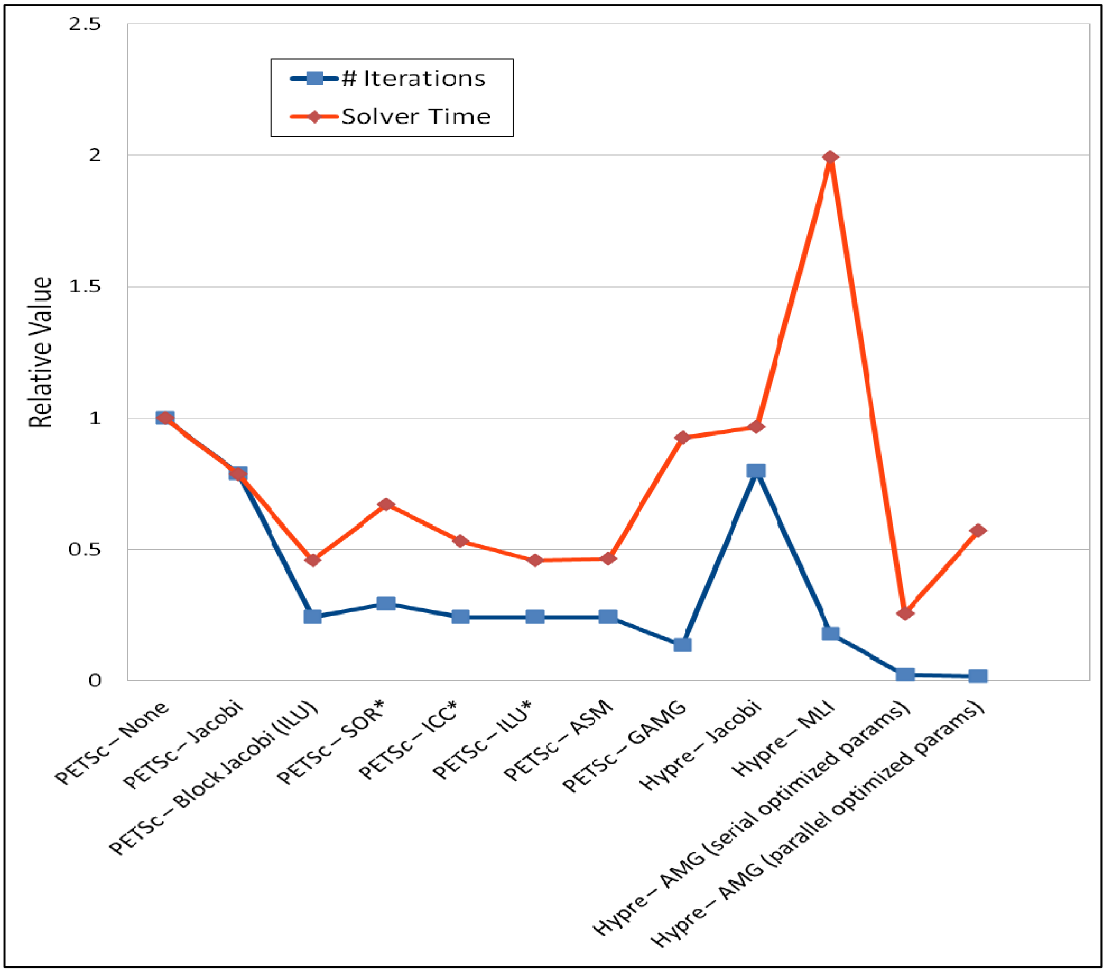

Figure 2. Relative serial performance of various solver – preconditioner combinations for mesh no. 2. Values are compared to the PETSc solver without preconditioning.

Table 3. Parametric study of the Hypre – AMG parameters to optimize serial performance. The bottom row corresponds to the final set of parameters referred to as "serial optimized params" throughout this work.

amgMaxLevels	amgCoarsenType	amgMeasureType	amgRelaxType	amgNumSweeps	amgRelaxWeight	amgRelaxOmega	amgStrongThreshold	amgSystemSize	amgMaxIterations	amgUseGSMG	amgGSMGNumSamples	Mesh #2	
												# Iterations	Solve Time [s]
Default (30)	falgout	Default (local)	*hybridsym*	1	Default (1.0)	Default (1.0)	0.9	1	Default (1)	No	default	183	178
Default (30)	falgout	Default (local)	*hybridsym*	1	Default (1.0)	Default (1.0)	0.9	1	Default (1)	*Yes*	default	393	286
Default (30)	falgout	Default (local)	*hybridsym*	1	Default (1.0)	Default (1.0)	0.9	3	Default (1)	No	default	70	66
Default (30)	falgout	Default (local)	*hybridsym*	1	Default (1.0)	Default (1.0)	0.9	*3*	Default (1)	Yes	default	51	38
Default (30)	falgout	Default (local)	*hybridsym*	2	Default (1.0)	Default (1.0)	0.9	*3*	Default (1)	*Yes*	default	42	48
Default (30)	ruge	Default (local)	*hybridsym*	2	Default (1.0)	Default (1.0)	0.9	*3*	Default (1)	*Yes*	default	42	48
Default (30)	Default (CLJP)	Default (local)	*hybridsym*	*1*	Default (1.0)	Default (1.0)	0.9	*3*	Default (1)	*Yes*	default	52	36
Default (30)	ruge3c	Default (local)	*hybridsym*	*1*	Default (1.0)	Default (1.0)	0.9	*3*	Default (1)	*Yes*	default	51	37
Default (30)	pmis	Default (local)	*hybridsym*	*1*	Default (1.0)	Default (1.0)	0.9	*3*	Default (1)	*Yes*	default	82	51
Default (30)	hmis	Default (local)	*hybridsym*	*1*	Default (1.0)	Default (1.0)	0.9	*3*	Default (1)	*Yes*	default	78	48
15	*Default (CLJP)*	Default (local)	*hybridsym*	*1*	Default (1.0)	Default (1.0)	0.9	*3*	Default (1)	*Yes*	default	52	37
45	*Default (CLJP)*	Default (local)	*hybridsym*	*1*	Default (1.0)	Default (1.0)	0.9	*3*	Default (1)	*Yes*	default	52	67
Default (30)	*Default (CLJP)*	Default (local)	*hybridsym*	*1*	Default (1.0)	Default (1.0)	1	*3*	Default (1)	*Yes*	default	51	36
Default (30)	*Default (CLJP)*	Default (local)	*hybridsym*	*1*	Default (1.0)	Default (1.0)	0.8	*3*	Default (1)	*Yes*	default	54	38
Default (30)	*Default (CLJP)*	global	*hybridsym*	*1*	Default (1.0)	Default (1.0)	*0.9*	*3*	Default (1)	*Yes*	default	52	38

Before examining the parallel efficiency of the various solvers and preconditions, we perform a second parametric study on the AMG parameters to determine the ideal set of parameters for parallel solves, as mentioned above. Figure 3 plots the solve time with increasing number of processing elements (PE) for the Hypre – Jacobi and Hypre – AMG using the "serial optimized params." These results show very poor scalability for AMG, with the solve time on eight PEs taking longer than on two PEs. We observe a crossover point at about 10 PEs where the AMG precondition takes more time than the Jacobi preconditioner for higher PE counts. To identify the ideal parameters for parallel solvers, we examine the solve time of mesh no. 2 on one PE and 32 PEs for various parameter combinations, as shown in table 4. The final set of parameters, listed on the last row of table 4 reduces the solve time on 32 PEs from 125.6 s for the serial optimized parameters to 4.9 s, a factor of 25 reduction in solve time. The parallel optimized parameters result in an increase to the solve time by a factor of two for one PE; however, with four or more PEs, the parallel parameters result in a lower solve time. The increased solve time with 128 PEs indicates that further parametric studies may be necessary to further optimize the parallel performance of the Hypre – AMG preconditioner. However, we aim to have approximately 5000–10,000 DOFs per PE for our final application, and therefore do not intend to distribute a mesh with the size of mesh no. 2 across more than approximately 64 PEs.

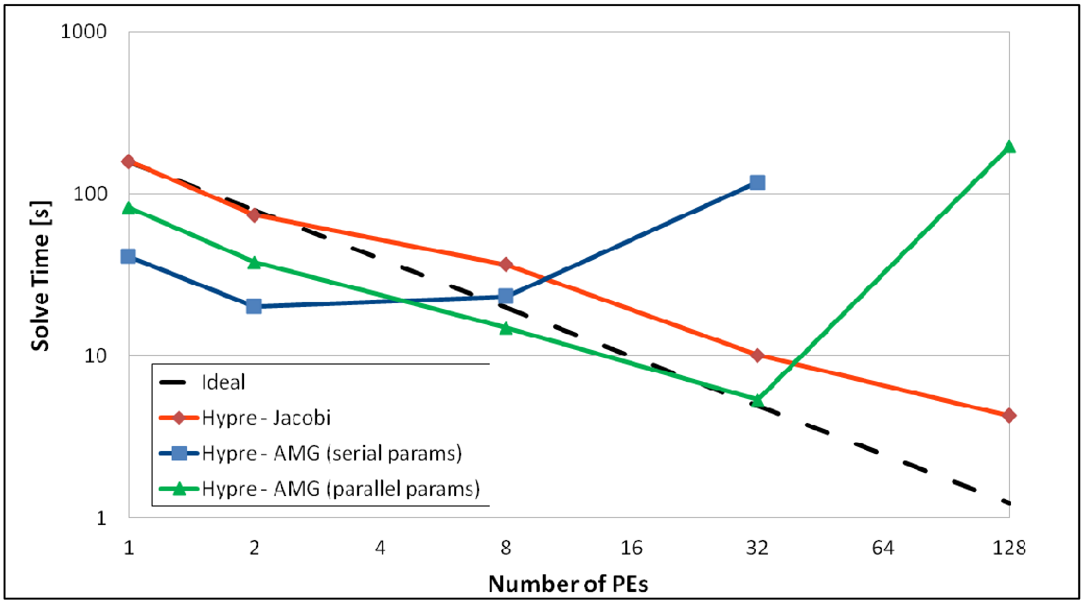

Figure 3. Parallel performance of Hypre – Jacobi and Hypre – AMG for two sets of AMG parameters. Ideal refers to the solve time of Hypre – Jacobi assuming 100% parallel efficiency.

Table 4. Parametric study of the Hypre – AMG parameters to optimize parallel performance. The bottom row corresponds to the final set of parameters referred to as "parallel optimized params" throughout this work.

amgMaxLevels	amgCoarsenType	amgMeasureType	amgRelaxType	amgNumSweeps	amgRelaxWeight	amgRelaxOmega	amgStrongThreshold	amgSystemSize	amgMaxIterations	amgUseGSMG	amgGSMG NumSamples	Mesh #2 (1 PE		Mesh #2 (32 Pes		Parallel Efficiency [%]
												# Iterations	Solve Time [s]	# Iterations	Solve Time [s]	
Default (30)	cljp	Default (local)	*hybridsym*	1	Default (1.0)	Default (1.0)	0.9	3	Default (1)	Yes	default	81	41.3	2889	125.6	1.03
Default (30)	cljp	Default (local)	*hybridsym*	1	Default (1.0)	Default (1.0)	0.9	3	Default (1)	No	default	94	73.5	197	12.6	18.20
Default (30)	cljp	Default (local)	*hybridsym*	1	Default (1.0)	Default (1.0)	0.6	3	Default (1)	No	default	50	62.5	102	9.5	20.62
Default (30)	cljp	Default (local)	*hybridsym*	1	Default (1.0)	Default (1.0)	0.3	3	Default (1)	No	default	37	77.5	165	21.7	11.17
Default (30)	cljp	Default (local)	*hybridsym*	1	Default (1.0)	Default (1.0)	0.45	3	Default (1)	No	default	41	62.8	154	15.5	12.63
Default (30)	cljp	Default (local)	*hybridsym*	1	Default (1.0)	Default (1.0)	0.75	3	Default (1)	No	default	66	63.9	143	11.0	18.17
Default (30)	cljp	global	*hybridsym*	1	Default (1.0)	Default (1.0)	0.6	3	Default (1)	No	default	50	61.9	102	9.1	21.30
Default (30)	ruge	global	*hybridsym*	1	Default (1.0)	Default (1.0)	0.6	3	Default (1)	No	default	51	59.3	225	23.1	8.01
Default (30)	falgout	global	*hybridsym*	1	Default (1.0)	Default (1.0)	0.6	3	Default (1)	No	default	50	59.5	136	13.3	13.98
Default (30)	ruge3c	global	*hybridsym*	1	Default (1.0)	Default (1.0)	0.6	3	Default (1)	No	default	51	59.1	271	25.4	7.27
Default (30)	pmis	global	*hybridsym*	1	Default (1.0)	Default (1.0)	0.6	3	Default (1)	No	default	144	97.2	924	43.0	7.06
Default (30)	hmis	global	*hybridsym*	1	Default (1.0)	Default (1.0)	0.6	3	Default (1)	No	default	127	85.9	703	34.7	7.74
Default (30)	cljp	global	*hybridsym*	1	0.1	Default (1.0)	0.6	3	Default (1)	No	default	100	159.3	100	11.2	44.61
Default (30)	cljp	global	*hybridsym*	1	1	0.1	0.6	3	Default (1)	No	default	86	136.4	92	9.8	43.64
Default (30)	cljp	global	*hybridsym*	1	1	0.5	0.6	3	Default (1)	No	default	54	86.3	57	5.1	52.76
Default (30)	cljp	global	*hybridsym*	1	1	0.75	0.6	3	Default (1)	No	default	51	81.0	56	5.3	48.10
Default (30)	cljp	global	*hybridsym*	1	1	0.85	0.6	3	Default (1)	No	default	50	79.6	65	6.0	41.28
Default (30)	cljp	global	*hybridsym*	1	1	0.25	0.6	3	Default (1)	No	default	62	97.4	66	6.3	48.59
Default (30)	cljp	global	*hybridsym*	1	1	0.65	0.6	3	Default (1)	No	default	52	82.9	55	4.9	52.41

Using the "parallel optimized params" for Hypre – AMG, we compare the parallel scalability of Hypre – AMG to other solver – preconditioner combinations with mesh no. 2 (figure 4) and mesh no. 3 (figure 5). In figure 4 we observe similar parallel scalability performance for each of the four solver – preconditioner combinations. The shortest solve times resulted from using Hypre –AMG and Hypre – Parasails, which have nearly identical timings along all PE counts. We observe nearly ideal scalability when increasing from one PE to eight PEs, but see the scalability drop to about 32% parallel efficiency (observed speed-up divided by ideal speed-up) when increasing to 64 PEs. Since the reduced parallel efficiency is observed across all solver – preconditioner combinations, it suggests that mesh no. 2 does not have enough DOFs to efficiently distribute across 64 PEs.

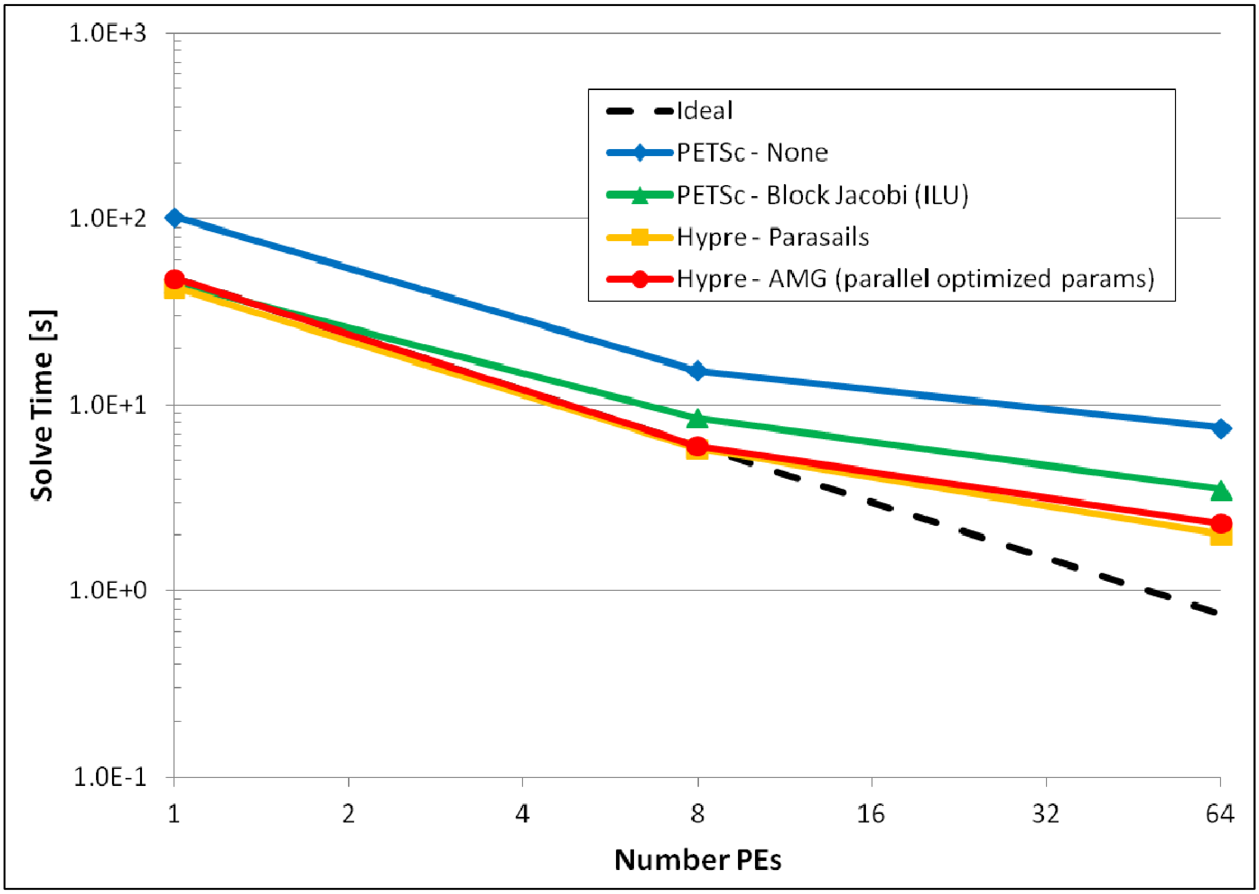

Figure 4. Parallel performance of various solver – preconditioner combinations with mesh no. 2. Ideal refers to the solve time of Hypre – AMG assuming 100% parallel efficiency.

When increasing the number of DOFs to over 22 million with mesh no. 3 (figure 5), we observe that excellent parallel efficiency can be achieved on most of the solver – preconditioner combinations for up to 1024 PEs. Despite having the worst parallel efficiency, Hypre – AMG has the smallest solve times across all PE counts. It appears that at some PE count above 1024, Hypre – Jacobi and/or PETSc – Block Jacobi may result in lower solve times than Hypre – AMG; however, further studies would be required for confirmation. We also note that Hypre was the only solver capable of solving mesh no. 3 on 64 PEs within the allotted time of one hour (h). The Linear CG solver ran out of time before completing the solve, while the PETSc solver ran out of memory while assembling the stiffness matrix. Because the focus of this study was on achieving the smallest solve times, the higher memory requirement of PETSc is not a significant concern; however, for memory constrained users, the lower memory usage of Hypre may become important.

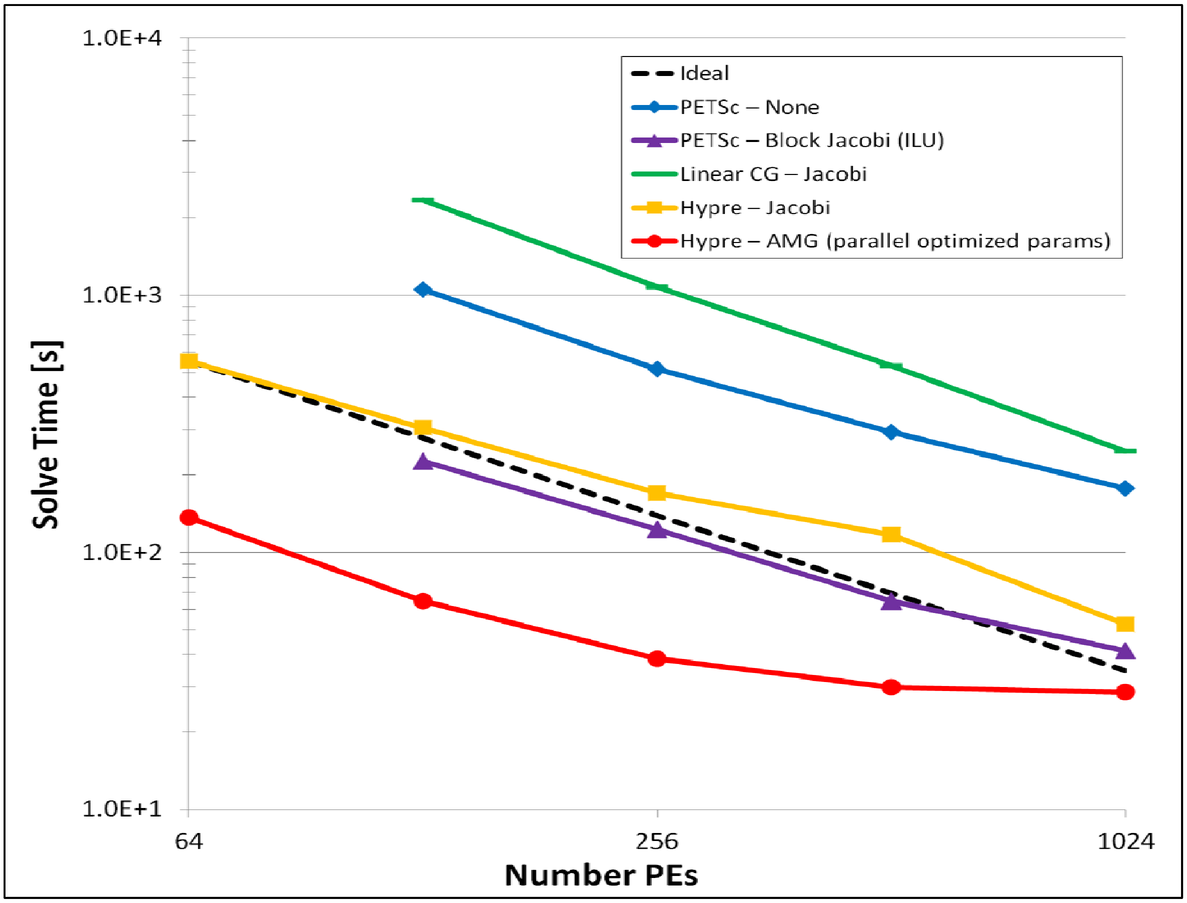

Figure 5. Parallel performance of various solver – preconditioner combinations with mesh no. 3. Ideal refers to the solve time of Hypre – Jacobi assuming 100% parallel efficiency.

4.2 Direct Solver

In applications where the coefficient matrix remains constant throughout a simulation, the use of direct solvers may become viable. The two primary methods utilized by direct solvers are to compute the inverse of the coefficient matrix and perform matrix-vector multiplication or compute a factorization of the coefficient matrix and perform back-substitution. The process of taking a matrix inverse or performing a complete factorization of a matrix is a resource-intensive operation, which scales as $O(n^3)$. However, once the inverse or factorization is complete for a given matrix, the solution for any given right-hand side vector can be determined quickly.

To determine the performance of a direct solver, we use MUMPS. In figure 6 we reproduce the parallel solve times for mesh no. 2 from figure 4 and add in the solve times for MUMPS. Since the factorization time can become prohibitively expensive with increasing system size, we also plot the setup time. These results indicate that the direct solver is as much as 60 times faster than the Hypre – AMG iterative method. While parallel speed-up is observed with MUMPS, we note that the parallel efficiency is worse than for any of the iterative methods. However, even on 64 PEs, we observe a factor of 20 reduction in solve time compared to Hypre – AMG.

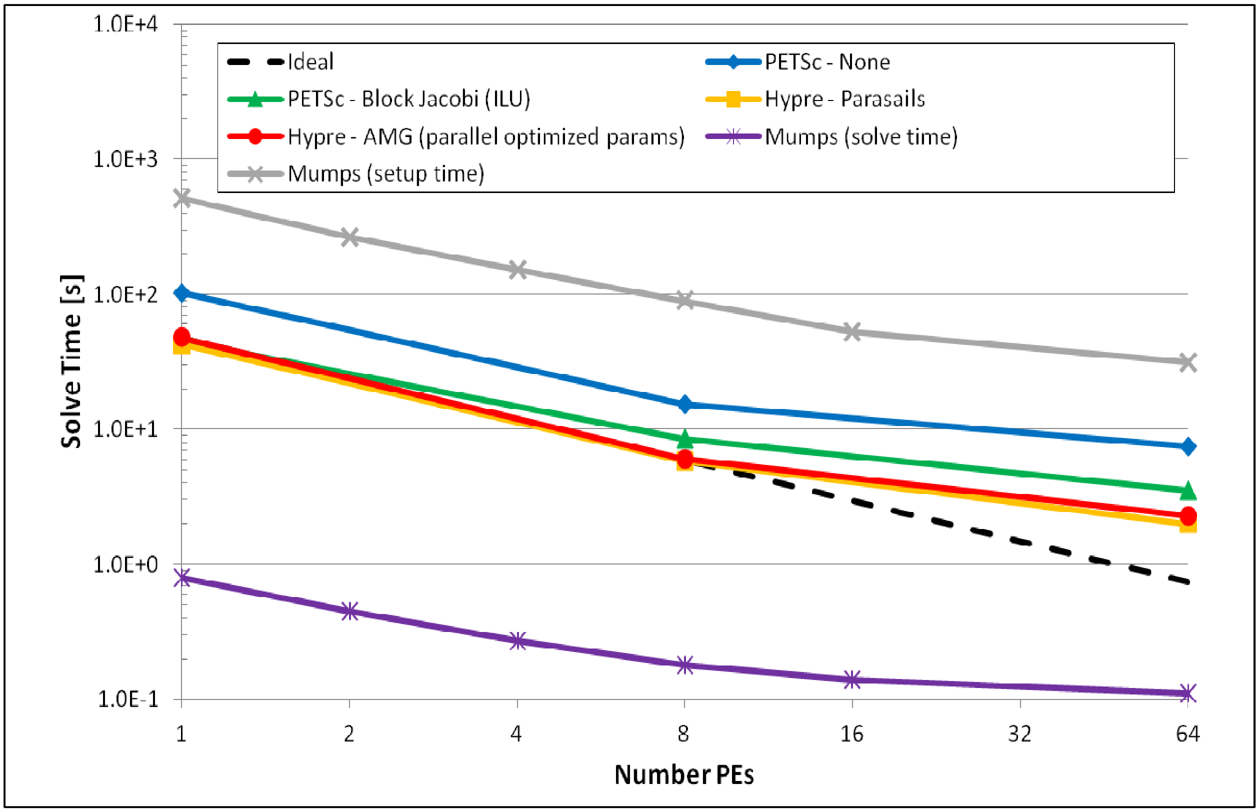

Figure 6. Comparison of direct solver with various iterative solver – preconditioner combinations. Both solve time and setup time for MUMPS are included. Ideal refers to the solve time of Hypre – AMG assuming 100% parallel efficiency.

To include the high setup costs when comparing MUMPS to Hypre – AMG, we define the metric *N* as the number of solves that need to be performed such that

$$T_{MUMPS}^{setup} + N\, T_{MUMPS}^{solve} < N\, T_{AMG}^{solve} \tag{7}$$

where T_{MUMPS}^{setup} is the wall clock time for MUMPS to perform setup and factorization, T_{MUMPS}^{solve} is the wall clock time for MUMPS to perform a solve, and T_{AMG}^{solve} is the wall clock time for Hypre – AMG to perform a solve. We ignore the Hypre – AMG setup time as it is small compared to the MUMPS setup time and we also assume the solve time is constant from one solve to the next. However, we note that including the Hypre – AMG setup time would only lower *N*, further improving the relative performance of MUMPS. The values of *N* for 1, 8, and 64 PEs are included in table 5.The results show that only 10–15 solves are required from MUMPS to recuperate the additional setup time. To put these savings of computational cost into perspective, we also compute the total wall clock time for Hypre – AMG and MUMPS if *N* = 1e5, which is the target number of timesteps we hope to achieve in FED3 simulations. We find that a one PE simulation that will take over 55 days with Hypre – AMG can be completed in less than a day with MUMPS. When scaling up to 64 PEs, a 64-h simulation is reduced to 3 h. These results show that utilizing a direct solver such as MUMPS can enable simulation time scales unattainable with iterative methods.

Table 5. Comparison of Hypre – AMG and MUMPS.

No. of PEs	N	Time for 1e5 Solves (h)	
		Hypre – AMG	MUMPS
1	10.89	1327.78	22.36
8	15.40	166.67	5.02
64	14.38	63.89	3.06

Despite the benefits of MUMPS observed with mesh no. 2, we found that MUMPS was unable to solve mesh no. 3 with any PE count up to 1024 due to memory limitations and setup times that went beyond a preset allotted time of 10 h. Simulations with intermediate mesh sizes revealed that MUMPS is able to handle approximately three million DOFs while keeping the number of DOFs on the order of 1e3–1e4 per PE. Therefore, large-scale simulations still require iterative methods.

5. Conclusions

In this report, we examine the performance of various linear solvers and preconditioners to identify the combinations with the shortest wall clock time for large-scale linear systems. Our results show that for system sizes of less than three million DOFs, direct solvers perform better as long as factorization of the coefficient matrix only needs to be performed once. With system sizes beyond three million DOFs, the factorization of a direct solver becomes intractable. For large systems, iterative solvers are required. We found that the best performing iterative method was the AMG preconditioner packaged in the Hypre library. However, the performance was sensitive to the AMG parameters selected. If a parametric study cannot be performed for a given system (or similar system), the Block Jacobi preconditioner packaged with the PETSc library showed good performance with the default preconditioning setting. The optimized AMG parameters determined in this work should be a good starting point for any 3-D linear elastic simulations, but transferability to other applications has not yet been studied. The choice of solvers and preconditioners can have a significant effect on the time required to solve a linear system of equations, and consequently the total simulation time. A number of factors should go into the selection of solver – preconditioner combinations including the desired system size, the size of the computing resources available, the reusability of preconditioning/factorization, and the number of solves to be performed in a given simulation.

6. References

Amestoy, P. R.; Duff, I. S.; L'Excellent. J. -Y. *MUMPS Multifrontal Massively Parallel Solver Version 2.0*, 1998.

Arsenlis, A.; Cai, W.; Tang, M.; Rhee, M; Oppelstrup, T.; Hommes, G.; Pierce, T. G.; Bulatov, V. V. Enabling Strain Hardening Simulations With Dislocation Dynamics. *Modelling Simul. Mater. Sci. Eng* **2007**, *15*, 553–595.

Baker, A. H.; Kolev, Tz. V.; Yang, U. M. *Algebraic Multigrid Interpolation Operators for Linear Elasticity Problems*; LLNL-JRNL-412928; 2009.

Balay, S.; Brown, J.; Buschelman, K.; Eijkhout, V.; Gropp, W.; Kaushik, D.; Knepley, M.; Curfman McInnes, L.; Smith, B.; Zhang, H. *PETSc Users Manual Revision 3.4*, 2013.

Crone, J. C.; Chung, P. W.; Leiter, K. W.; Knap, J.; Aubry, S.; Hommes, G.; Arsenlis, A. A. Multiply-Parallel Implementation of Finite Element-Based Discrete Dislocation Dynamics for Arbitrary Geometries. *Modelling Simul. Mater. Sci. Eng*., 2013, in preparation.

Falgout, R. D.; Yang, U. M. *Hypre: A Library of High Performance Preconditioners*, Computational Science – ICCS (Springer), 2002; 632–641.

Hughes, T. J. R. *The Finite Element Method,* Dover Publications, Inc., Mineola, NY, 2000.

Shewchuk, J. R. *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*, Carnegie Mellon University, Pittsburgh, PA, 1994.

List of Symbols, Abbreviations, and Acronyms

3-D	three-dimensional
ANL	Argonne National Laboratory
ARL	U.S. Army Research Laboratory
BVP	boundary value problem
CG	conjugate gradient
DDD	discrete dislocation dynamics
DDD-FEM	discrete dislocation dynamics-finite element method
DOD	Department of Defense
DOF	degree of freedom
DSRC	DOD Supercomputing Resource Center
FEM	finite element method
h	hour
LLNL	Lawrence Livermore National Laboratory
MUMPS	Multifrontal Massively Parallel Sparse Direct Solver
PE	processing elements
PETSc	Portable, Extensible Toolkit for Scientific Computation
s	second
SPD	symmetric positive definite

NO. OF COPIES	ORGANIZATION
1 (PDF)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA
1 (PDF)	DIRECTOR US ARMY RESEARCH LAB IMAL HRA
1 (PDF)	DIRECTOR US ARMY RESEARCH LAB RDRL CIO LL
1 (PDF)	GOVT PRINTG OFC A MALHOTRA
1 (PDF)	RDRL CIH C J CRONE

INTENTIONALLY LEFT BLANK.